Escuela Politécnica Superior

Sistema de detección de obstáculos aéreos para invidentes DIY

Grado en Ingeniería en Sonido e Imagen en Telecomunicación

Trabajo de Fin de Grado

Autor:
Pedro Patón Valentín, DNI: 71721358K
Tutor/es:
Juan Manuel Martínez Sáez

Junio 2018

Universitat d'Alacant
Universidad de Alicante

Agradecimientos

Este proyecto ha sido fruto de una gran carga de trabajo y preparación durante el último año, el cual ha estado lleno de presión de esfuerzo y de dedicación, es por eso que debo hacer una mención especial a varias personas para agradecerles lo que me han aportado.

En primer lugar agradecer a **Juan Manuel Martínez Sáez** y a la **Universidad de Alicante** la oportunidad única y los materiales para poder comenzar este proyecto que para mí es especial, además de todo lo que me ha aportado mi tutor Juan Manuel por su paciencia y su ayuda para poder corregir, completar y conseguir cada parte de este proyecto.

En segundo lugar a **mis padres y mi hermano** por su apoyo incondicional, pero en especial a **Joaquín Patón Pardina**, mi padre, por ayudarme a la parte técnica y compartir sus conocimientos para darle un toque más profesional además de completar este proyecto con ideas y soluciones que de haberlas hecho solo me habría costado muchísimo más.

Por último lugar mencionar a una persona que me dejo durante este curso, **mi abuelo Pedro**, ya que pese al dolor de su muerte conseguí encarrilar en forma de motivación para hacer las cosas mejor y esforzarme al máximo para hacer algo de lo que seguro él estaría orgulloso, ya que pese a tener 98 años y no haber tenido la posibilidad de estudiar, admiraba la tecnología me apoyaba y le encantaba que le explicásemos todos los avances tecnológicos.

En cuanto a una reflexión personal sobre el trabajo, a la hora de realizar uno, buscaba un proyecto que fuese distinto, una alternativa que pudiese, de algún modo servir a alguien con un problema real en su vida cotidiana.

Juan Manuel, mi tutor, me dijo que pensase en algún tipo posible aplicación que utilizase procesado de sonido y que ayudase a solventar los problemas diarios de una persona invidente.

Desde ese momento, fui involucrándome en cómo una persona invidente vive su día a día, las dificultades, las actividades que para cualquier persona son una tarea fácil y que para ellos, al no disponer del sentido de la vista, puede resultar una hazaña.

A pesar de ser un año completo y difícil, he disfrutado muchísimo aprendiendo un nuevo lenguaje como es el de Arduino, y a programar con un sistema que no tiene interrupciones con su consecuente dificultad. Pero como antes he citado, al ser un proyecto que busca el bien de las personas invidentes, el resultado positivo de este trabajo es una gran recompensa, al saber que con un poco de esfuerzo se puede facilitar tanto la vida a alguien.

Por otra parte quería destacar las infinitas posibilidades, que este proyecto puede llegar a alcanzar gracias a la comunidad de internet, y a la posibilidad de que alguien

desinteresadamente pueda mejorar la solución que nosotros obtenemos, aportando su granito de arena.

Es por todas estas razones que con este proyecto solo puedo destacar el balance positivo, que me queda tanto como estudiante que constantemente está aprendiendo, como por la parte de persona por la faceta humana de saber que es posible ayudar a alguien con tu trabajo.

Resumen

En este proyecto, se quiere diseñar un sistema detector de obstáculos aéreos.

Los obstáculos aéreos son aquellos que se encuentran por encima del pecho hasta la cabeza de las personas. Estos, tienen especial problemática al no poder ser detectados por las principales herramientas con las que normalmente se orienta una persona invidente, como son los perros guía, que no son capaces de discernir la altura del objeto en comparación con la de su dueño, o los bastones, los cuales principalmente detectan los obstáculos terrestres.

Además, se busca obtener una solución económica, que mediante el uso de la tarjeta Arduino, consiga detectar obstáculos aéreos, alertando a la persona que lo utiliza con estímulos sonoros y hápticos. De esta manera se cubrirían las necesidades tanto de discapacitados visuales, como de invidentes que además sean sordos.

Otro punto a favor de la inclusión de esos dos tipos de estímulos, sería la posibilidad de que una persona ciega no quisiera llamar la atención, o que necesitara estar en un lugar donde el silencio fuera imperioso, de esta forma podría integrarse sin sentirse discriminada.

La solución propuesta, es un dispositivo “wearable”, en este caso, una gorra. Dicha gorra tendrá, 3 sensores de ultrasonidos dispuestos en la visera, 3 motores vibradores en la zona interior de la gorra que está en contacto con la cabeza, un zumbador dispuesto en la parte de debajo de la visera que está detrás de los sensores, la tarjeta Arduino y una batería para alimentar todo el sistema. Todos los componentes, estarán conectados mediante los cables pertinentes.

Este proyecto tiene la particularidad de ser DIY, en inglés “Do It Yourself”, esto significa que mediante un listado de componentes, los dos códigos que se han programado y unas instrucciones de montaje, cualquier persona que no tenga necesariamente un conocimiento avanzado en la materia, pueda fabricarse esta solución por un precio económico.

Además, este tipo de proyectos, dota a la solución de un futuro prometedor, gracias a la gran comunidad de internet, que facilita una realimentación, tanto por parte de la gente que lo pruebe, como por parte de posibles programadores experimentados o que tengan un amplio conocimiento en la materia, para mejorar en cualquier ámbito el proyecto.

Índice:

1. Justificación y objetivos

1.1 Contextualización del problema

Los obstáculos aéreos, son aquellos que no tienen proyección sobre el suelo, están situados por encima del pecho y que además pueden causar daños físicos o desorientación a la persona invidente. Estos, tienen especial problemática, porque no pueden ser detectados por las herramientas cotidianas de las personas invidentes, como puede ser un bastón, que se utiliza haciendo barridos del suelo, o un perro guía, ya que el perro no es capaz de discernir la altura de su amo, conforme a la posición de los obstáculos aéreos que haya cerca de él.

Figura 1 y 2. Ejemplo de obstáculos aéreos

En segundo lugar la ceguera está considerada como la principal discapacidad de los sentidos, ya que más del 80% de la información sensorial del entorno que tenemos los humanos viene gracias a ella.

Es por ello, que las personas con cualquier tipo de discapacidad visual (ya sea ceguera completa, parcial o cataratas) necesitan una especial ayuda para ser integradas y que puedan tener una interacción lo más completa posible con su entorno y la sociedad.

En tercer lugar, los costes de fabricación de cualquier dispositivo electrónico que sirva a este colectivo son muy altos, esto es debido al reducido número de discapacitados visuales que podrían adquirir dichos productos. Según la ONCE sólo 72.240 personas de los 46 millones y medio de españoles están reconocidas legalmente como invidentes en el año 2017.

1.2 Dispositivos y soluciones tecnológicas actuales

Para poder obtener un análisis de los sistemas o prototipos detectores de obstáculos aéreos que actualmente están disponibles. Hay que desgranar: su funcionalidad, la tecnología que utilizan y características, que en función con el precio, nos ayudarán a plantear los objetivos que buscamos cumplir con el proyecto.

1.2.1 Sunu Band

Este dispositivo creado por la empresa **SUNU,** es una pulsera provista de un sensor de ultrasonidos, y utiliza una ecolocalización sonar, que traduce la distancia en vibración háptica.

En segundo lugar, puede utilizarse para ponerla en el mango del bastón, orientada en diagonal, para que de esa manera detecte los obstáculos aéreos.

Tiene un precio de 299 dólares, tal importe es excesivamente alto a mi parecer. Pero por todo lo demás es una solución muy discreta y sencilla de usar, lo que al usuario beneficia bastante. Por último destacar que esta empresa hace envíos a cualquier parte del planeta.

Figura 3. Sunu band de la marca Sunu

1.2.1 Sonar Globe

Este dispositivo es un proyecto creado por **DavidDrone**, un usuario de la página **Instructables.com.** Su solución es simple, se trata de un guante con un sensor de ultrasonidos, que al detectar un obstáculo posicionado delante del usuario, avisa con un testigo sonoro. Es una solución creada mediante el empleo de una tarjeta **Arduino.**

La solución es muy parecida a **Sunu band**, la única diferencia, es que no es necesario que se adapte al bastón, ya que se puede orientar a donde se quiera. El punto negativo, es que puede ser incómoda al tener que mantener la mano en una posición concreta.

En cuanto al factor económico se acerca al tipo de solución que queremos dar, ya que con unos 60 € y el software que él proporciona puede crearse dicho sistema de detección.

Figura 4. Prototipo Sonar Globe

1.2.2 Eyesynth

Eyesynth, es un sistema de detección de obstáculos y comprensión visual, para invidentes, basado en visión estereoscópica.

La visión estereoscópica, es una tecnología que utiliza dos cámaras para obtener una imagen en tres dimensiones del entorno. En el caso de **Eyesynth**, transforma esta imagen 3D en un mapa sonoro.

Este producto, está compuesto por unas gafas, que registran el espacio en el que está el sujeto en tres dimensiones y un microrordenador que procesa los datos captados por las lentes, para convertirlo en audio que pueda guiar a la persona que lo use.

Figura 5. Componentes de Eyesynth

Las 3 características técnicas fundamentales de esta solución son:

- **Funcionalidad 3D:** El usuario es capaz de detectar no solo objetos y formas, sino que además, puede detectar profundidad para localizar todo con más precisión.
- **Sonido abstracto:** Utiliza un nuevo lenguaje abstracto, que es fácil de asimilar y automatizar por el cerebro.
- **Audio coclear:** Conduce el sonido a través de los huesos de la cabeza. Esta característica es muy importante, ya que así, no se inhibe la audición, además de que reduce la fatiga por un prolongado uso del Eyesynth.

Las características más importantes del software, son los dos modos de funcionamiento**: Rastreo** (el usuario, gira con el cuello para barrer y así poder reconocer la zona central, de manera análoga a cuando se utiliza un bastón) **y Panorama completo** (El usuario obtiene los estímulos sonoros que describen el entorno sin necesidad de girar la cabeza para detectar cada obstáculo). Además pueden seleccionarse distancias de medida, entre las que se pueden seleccionar 0.7, 2 y 6 metros. Por último destacar que tiene un testigo sonoro para alertar del nivel de batería.

En resumen, este sistema es una solución brillante que puede conseguir la completa autonomía de una persona invidente. Por otra parte la gran desventaja de este sistema es el precio, 2420 €, lo que lo hace una alternativa inviable para un gran número de personas.

1.2.3 DOA utilizando móviles con cámaras 3D

Este proyecto, presentado el 7 de mayo de 2014, fue una solución para la detección de obstáculos aéreos, utilizando la visión estereoscópica de la misma manera que **Eyesynth**, explotando el hecho de que los móviles empezaron a comercializarse con 2 cámaras; herramienta que utilizaron para crear un algoritmo capaz de trazar a un mapa en tres dimensiones.

Con las dos cámaras es capaz de obtener la **distancia focal** en pixeles y **la línea basal** en metros. Gracias a esa información, crean un histograma de distancias, lo cual hace que se puedan detectar la distancia a la que está el obstáculo de enfrente.

Todo esto llevado a cabo en una aplicación móvil, hacía que fuese una gratuita a la par que novedosa alternativa para la detección de obstáculos aéreos.

Figura 6. Interfaz aplicación DOA

El problema de este proyecto fue, que los diseñadores de teléfonos dejaron de implementar la tecnología de doble cámara, debido a que abarataba el coste y en esa época no aportaban mucha utilidad en cuanto a las aplicaciones móviles se refiere.

1.2.4 UltraCane

Es posiblemente el sistema más conocido actualmente para la detección de obstáculos aéreos. Está basado en un sistema de ultrasonido de haz estrecho de doble rango que proporciona un 100% de previsión de obstáculos, tanto aéreos como terrestres que estén a una distancia superior a la que un bastón pueda llegar.

Figura 7. UltraCane

Utiliza dos sensores de ultrasonidos que proporcionan datos sobre el rango en el que se encuentran los obstáculos, como ramas de árboles, mobiliario urbano, personas, etc.

Esta solución, puede ser usada tanto en entornos cerrados, como en entornos abiertos, dando un nivel de independencia al usuario invidente muy alto.

Los estímulos que proporciona para informar al sujeto que utiliza el sistema son vibraciones, enviados a través de dos pequeños botones situados en el mango del bastón.

Este bastón es un gran aliado para la persona invidente, pero tiene en contra su precio, el cuál es de 635 libras, lo que son un total de 724 € más o menos. Esto lo hace una alternativa más económica que el Eyesynth, aunque menos tecnológica.

Sea como fuere, es un sistema, que se queda muy alejado económicamente para que cualquier persona pueda tener acceso a él.

1.3 Objetivos

Una vez que hemos analizamos el problema en profundidad, el mercado y todo lo que pueden ofrecer a los usuarios invidentes, procedemos a establecer los objetivos principales de este proyecto:

- **Proyecto DIY**: El proyecto se hará, para que la gente pueda descargarse nuestras instrucciones de montaje y el código. Así con un presupuesto **económico** podría tener nuestra solución accesible toda persona con cualquier tipo de discapacidad visual.
- Implementar un **hardware específico** para solventar el problema de los obstáculos aéreos, que **complemente a las herramientas** principales que las **personas invidentes** utilizan para guiarse.
- **Wearable**: El proyecto tendrá una estructura que le permita ser llevado como prenda de ropa, con el fin de que el usuario pueda utilizarlo como un complemento de uso cotidiano.
- **Uso de un ángulo de cobertura que cubra un rango que permita autonomía al invidente**: Para que la solución sea más completa, se utilizarán los sensores de ultrasonidos pertinentes, que además deberán tener una localización estratégica para poder satisfacer la correcta localización de obstáculos aéreos.
- **Uso de 2 tipos de estímulos**: Por cuestión de integridad psicológica de la persona, para que esta solución pueda ser utilizada en entornos donde el silencio es imperioso o en ocasiones que puedan avergonzar al usuario (por ejemplo en un auditorio), además de que este sistema pueda ser utilizado por personas que a la vez de invidente sean sordas; se programarán dos códigos: Uno que emitirá testigos sonoros mediante un zumbador y otro que utilizará la vibración para orientar al usuario, ambos evitando que entorpecimientos del entorno entren en conflicto con la parte superior del cuerpo del invidente.
- **Rango suficiente para la previsión de choque**.
- **Discriminación de la distancia para informar al usuario**: Para ello el código debe estar preparado de modo que, el prototipo cambie la frecuencia tanto del zumbador, como de los vibradores, según se acerque más o menos un obstáculo. Además debe discernir para cada uno de los 3 sensores, asignándoles una frecuencia en el caso del zumbador o cada uno de los tres vibradores dependiendo de qué sensor de ultrasonidos esté activado.

2. Propuesta de Solución del problema

2.1 Arduino

2.1.1 Hardware

Para cumplir con todos los objetivos que nos habíamos propuesto, necesitábamos una solución, que además de potente, fuese económica e intuitiva para montar. Es por ello que escogimos el hardware Arduino.

Arduino es un tipo de microcontrolador de código abierto y software libre, con una capacidad abrumadora de creación y desarrollo de códigos y aplicaciones, diferentes tipos de sensores.

En este caso, se ha utilizado el modelo UNO (de Arduino), las características más destacables de esta tarjeta es que tiene 14 pines digitales con 6 salidas PWM para alimentación de motores, 6 entradas analógicas, 32k de memoria flash y un reloj interno que trabaja a 16Mhz.

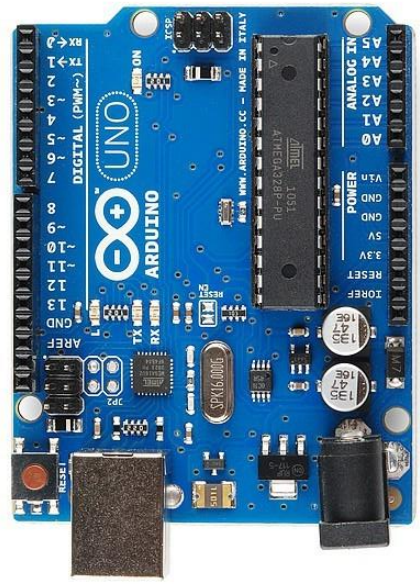

Figura 8.Tarjeta Arduino UNO

Arduino fue creado por el instituto IVRAE, que más adelante pasó a llamarse grupo Arduino (de procedencia italiana), su lenguaje de programación es C, con un gran abanico de librerías tanto predeterminadas, como creadas por la comunidad que son bastante intuitivas.

En cuanto a sus posibilidades son infinitas, ya que tiene una cantidad ingente de sensores y complementos, que unidos a un buen código pueden hacer que se consiga

soluciones para casi cualquier problema, además que todas ellas serán muy económicas.

En segundo lugar escogeremos sensores de ultrasonidos para calcular la distancia, más concretamente, tres. Los sensores antes mencionados serán el modelo HC-SR04, éstos tienen 4 patillas: una para la alimentación (VCC), otra para la tierra (GND), y otras dos que son Trig y Echo, que sirven para enviar la señal que activará el pulso y para recibir la señal convertida en impulso eléctrico respectivamente, lo cual nos ayudará a calcular la distancia a la que está el obstáculo.

Figura 9. Sensor de ultrasonidos HC-SR04

En tercer lugar para la obtención de estímulos para el usuario, y que, como ya comentamos en los objetivos, sirva para diferentes situaciones y para personas que además de invidentes sean discapacitadas auditivas.

Utilizaremos dos tipos de elementos:

- El primero es un zumbador, componente que es capaz de producir un tono. Lo utilizaremos para que cree una secuencia de tonos para poder discernir solo con una unidad, el posicionamiento del obstáculo, además de su cercanía al sujeto, que porta nuestra solución.

Figura 10. Ejemplo de zumbador

- El segundo que usaremos será un vibrador; para ser más exactos utilizaremos 3, uno para cada lugar de donde pueda proceder la alerta de obstáculo (izquierda, derecha o centro). Por último comentar que dichos vibradores aumentarán su frecuencia conforme el impedimento se acerque al usuario.

Figura 11. Ejemplo de vibrador

Para terminar, comentar que la idea principal es crear dos códigos, los cuales funcionarán uno con estímulos auditivos (zumbador), y otro con estímulos hápticos (vibrador).

2.1.2 Software

El software utilizado es un compilador que tiene las librerías que maneja Arduino, puede descargarse la aplicación o compilarse online, funciona en dos tiempos uno compilar y ver que no tiene errores y en el segundo sube el código a la tarjeta.

Tiene además una representación tanto con valores como en gráficas de los valores que le indiques con las instrucciones **Serial.plot** y **Serial.Write**.

2.1.3 Ejemplos de códigos

Son muchos los componentes que tiene Arduino para programar y crear prototipos, esta es una lista de códigos que utilizan los componentes escogidos para llevar a cabo el prototipo del detector de obstáculos aéreos.

2.1.3.1 *Sensor de Ultrasonidos:*

El circuito está basado en este tipo de sensor, lo que hace es que, en una pantalla disponible en el compilador, saque por pantalla, según un cálculo de una teoría de eco, la distancia a la que estaba el obstáculo.

Esquema:

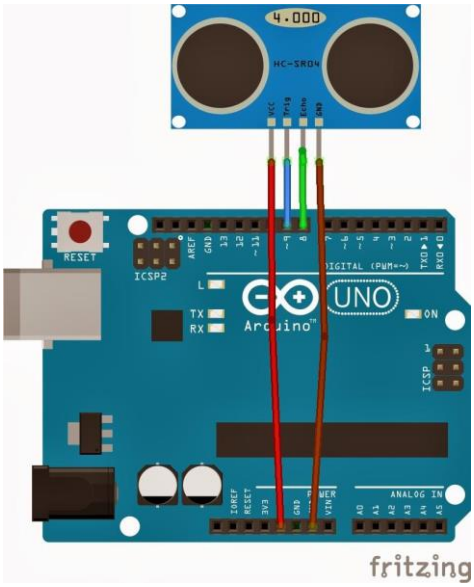

Figura 12.Esquema circuito Sensor de ultrasonidos

Código:

```
long distancia;//variables para calcular según lo que
devuelve el sensor
long tiempo;
void setup() {
    Serial.begin(9600);
    pinMode(9, OUTPUT);
    pinMode(8, INPUT);
}
void loop() {
    // put your main code here, to run repeatedly:
   digitalWrite(9,LOW); /* Por cuestión de  estabilización
   del sensor*/
    delayMicroseconds(5);
    digitalWrite(9, HIGH); /* envío del pulso ultrasónico*/
    delayMicroseconds(10);
   tiempo=pulseIn(8, HIGH); /* Función para medir la
   longitud del pulso entrante. Mide el tiempo que
   transcurrido entre el envío del pulso ultrasónico y
   cuando el sensor recibe el rebote, es decir: desde que el
   pin 8 empieza a recibir el rebote, HIGH, hasta que deja
   de hacerlo, LOW, la longitud del pulso entrante*/
   distancia= int(0.017*tiempo); /*fórmula para calcular la
   distancia obteniendo un valor entero*/
    /*Monitorización en centímetros por el monitor serial*/
```

```
    Serial.println("Distancia ");
    Serial.println(distancia);
    Serial.println(" cm");
    delay(1000);
}
```

Explicación:

La primera parte del código es añadir dos variables tipo long (64 bits con signo), que serán tiempo y distancia, las cuales serán usadas en una fórmula para sacar por pantalla la distancia a la que se encuentra el obstáculo. Por último en esta parte definiremos el pin 9 como salida y el 8 como entrada.
Seguidamente por temas de estabilización de sensor usaremos la función **delayMicroseconds** para hacer un delay de 5 ms y pondremos el pin 9 a **LOW**. El siguiente paso es enviar el pulso ultrasónico poniendo el pin 9 a **HIGH**, la variable **tiempo,** vendrá definida por el cálculo mediante el cual la función **pulseIn** mide el tiempo pasado desde que el pin 8 recibe el rebote del pulso hasta que deja de recibirlo.
Por último escribirá distancia por la pantalla del compilador con **Serial.println**, distancia es calculada con 0.017*tiempo.

2.1.3.2 Zumbador:

Este circuito consta de un zumbador que emite la secuencia de una escala musical, además logré hacer varias sinfonías cortitas para su funcionamiento.

Esquema:

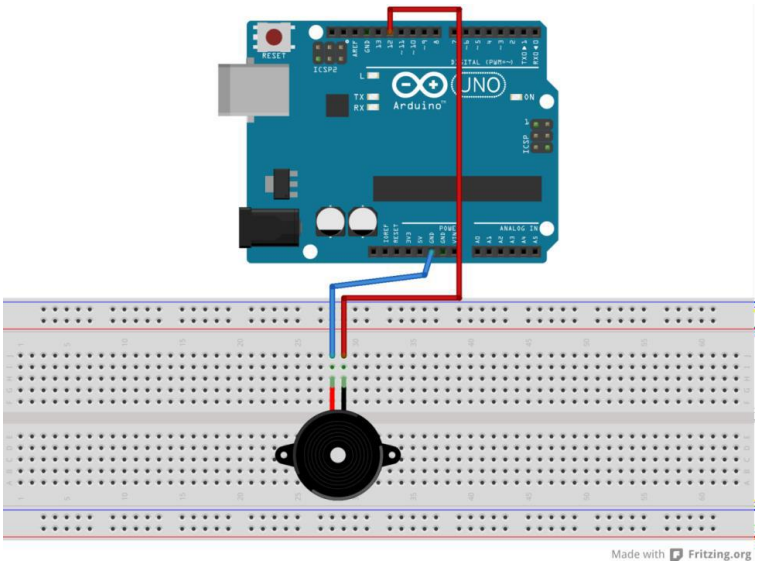

Figura 13.Esquema circuito Zumbador

Como el funcionamiento de un zumbador y un vibrador es el mismo, probé el código con un vibrador, al cual tuve que añadirle regletas por la delgadez de los cables y funcionaba de la misma manera que el zumbador, el problema es que no

le daba tiempo a hacer las iteraciones y casi no se notaba el cambio de intensidad de vibración, la solución sería utilizar un rango de frecuencias mucho menor y más espaciadas entre los valores de éstas.

Código1:

```
// Introducimos la variable por donde saldrá nuestra señal
digital hasta el zumbador
int speakerPin = 12;
// Definimos una variable con el número de tonos que va a
reproducir
int numTones = 10;
int tones[] = {261, 277, 294, 311, 330, 349, 370, 392, 415,
440};
void setup() {
}
void loop() {
    // Generamos un bucle que recorra nuestro vector. Este
    será el encargado de introducir una determinada
    frecuencia al zumbador cada vez, conforme hayamos
    declarado el vector de tonos.
     for (int i = 0; i < numTones; i++)
     {
         tone(speakerPin, tones[i]);
         delay(500);
     }
     noTone(speakerPin);
}
```

Explicación:

El funcionamiento de este código comienza con definir el pin del zumbador, después el número de iteraciones que hará, en este caso valdrá 10 debido a que tenemos 10 notas en el vector para hacer una escala.

Por último en el bucle principal habrá un for de "**n**" iteraciones que con la función **tone** enviará una señal por el pin 12 para que suene el zumbador en cada iteración a una frecuencia diferente, resultando una escala con un delay de 500ms entre cada nota, para que se apague el zumbador o el vibrador está la función **noTone** al final del recorrido del bucle principal.

Otro ejemplo de código que hice, fue la melodía de Blancanieves, el funcionamiento es el mismo, la única diferencia es que no hay bucle for, solo hay funciones **tone** con la repetición de las frecuencias y un delay entre cada nota:

Código2:

```
// Introducimos la variable por donde saldrá nuestra señal
digital hasta el zumbador
int speakerPin = 12;
void setup() {
}
```

```
void loop() {
   // put your main code here, to run repeatedly:
    // Generamos un bucle que recorra nuestro vector. Este
    será el encargado de introducir una determinada
    frecuencia al zumbador cada vez, conforme hayamos
    declarado el vector de tonos.
     tone(speakerPin, 392);//sol
     delay(500);
     tone(speakerPin, 400);//la
     delay(500);
     tone(speakerPin, 392);//sol
     delay(500);
     tone(speakerPin,350);//fa
     delay(500);
     tone(speakerPin,330);//mi
     delay(500);
     tone(speakerPin,350);//fa
     delay(500);
     tone(speakerPin, 392);//sol
     delay(500);
     noTone(speakerPin);
}
```

Explicación:

Este fue otro ejemplo de código que hice, es la melodía de Blancanieves, el funcionamiento es el mismo, la única diferencia es que no hay bucle for, solo hay funciones **tone** con la repetición de las frecuencias y un delay entre cada nota:

2.2 Proposición de diseño

Nuestra proposición es el diseño de una gorra, la cual tendrá 3 ultrasonidos posicionados en la visera, 3 vibradores en la zona interior que roza con la cabeza y el zumbador localizado en la parte posterior de la visera. Además tendrá una batería conectada a la tarjeta, para que alimente con el amperaje necesario la placa y todos los sensores.

En cuanto a dónde exactamente se colocan los ultrasonidos, debemos tener en cuenta, las características técnicas de estos. Las más destacables son su rango máximo de medida, 4 metros, y el ángulo de medición, que son 15 grados.

Esto condicionará la separación entre los sensores, ya que mide 15 grados para cada uno de los horizontales y otros 15 para los verticales. Por lo tanto debe haber una separación suficiente para que no se obstaculicen (en total de suma de rango de izquierda a derecha).

Para la sujeción de estos, se utiliza unas piezas de aluminio recortadas para hacer que se mantenga firme y sujeto frente a cualquier imprevisto. Estarán afianzados mediante un tornillo que llevará una pizca de silicona para que no se caiga el sensor.

Figura 14. Proposición para la sujeción de los ultrasonidos en la gorra

Figura 15. Proposición para la disposición de los ultrasonidos en la gorra

En segundo lugar para tener bien posicionados los vibradores, se coserán 3 bolsillos en unas posiciones análogas a las que están los sensores de ultrasonidos, en la zona de la gorra que están en contacto con la cabeza.

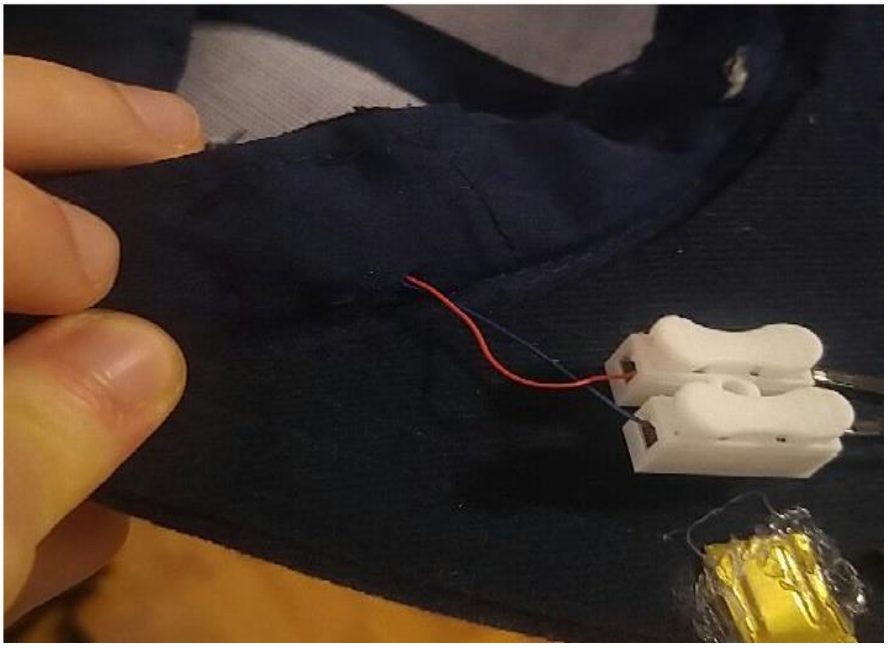

Figura 16.Proposición para la disposición de los vibradores en la gorra

Por último se colocará el zumbador en la visera, detrás de los sensores de ultrasonidos, éste, irá sujeto adecuadamente para garantizar que no se caiga por cualquier casualidad.

Figura 17. Proposición para la disposición del altavoz

2.3 Solución Propuesta

La idea principal de este proyecto es que se calcule la distancia con los sensores de ultrasonidos y que para alertar al usuario, haya dos tipos de estímulos: **auditivos y hápticos**.

Para el primer tipo de estímulos, el sistema debe calcular la mínima distancia a la que se encuentran los ultrasonidos que están por debajo del umbral de 2 metros, para posteriormente, emitir una secuencia de tonos identificada con el ultrasonido que ha medido esa distancia mínima y que conforme esté más cerca el obstáculo aéreo, la frecuencia de la secuencia sea mayor.

Para el segundo tipo de estímulos, se utilizarán 3 vibradores, los cuales estarán relacionados con cada uno de los 3 sensores, y que vibrarán con mayor o menor frecuencia dependiendo de la distancia a la que esté el obstáculo, siempre teniendo en cuenta el umbral, que en este caso proponemos que sean 2m.

Una vez descrita la idea de código, procederemos a describir lo como están conectados los componentes.

En primer lugar dispondremos los pines de tal modo, que se puedan conectar todos los aparatos de una manera correcta sin que se estorben y solo haga falta cambiar el código en la tarjeta para usar cada uno de los estímulos.

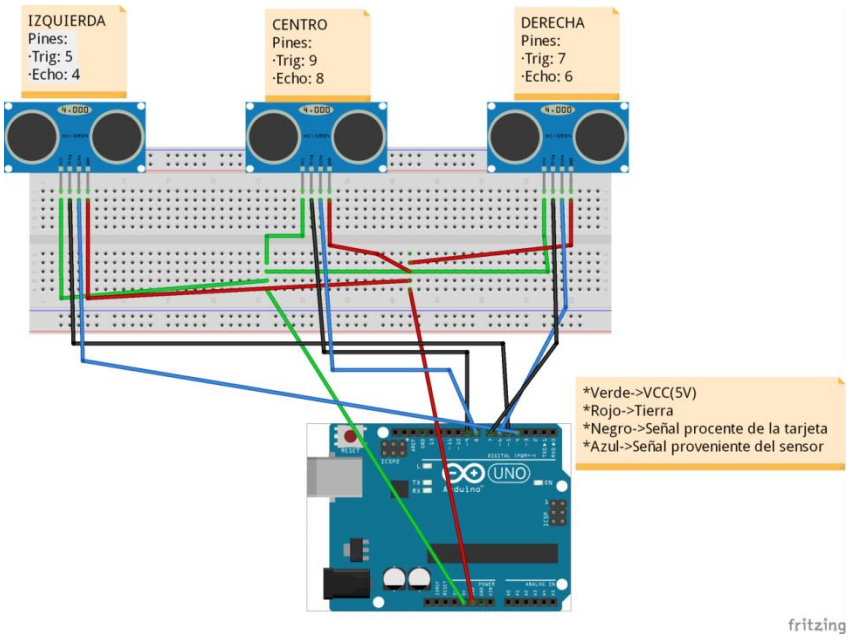

Figura 18. Esquema de conexión de los sensores de ultrasonidos en los pines de Arduino

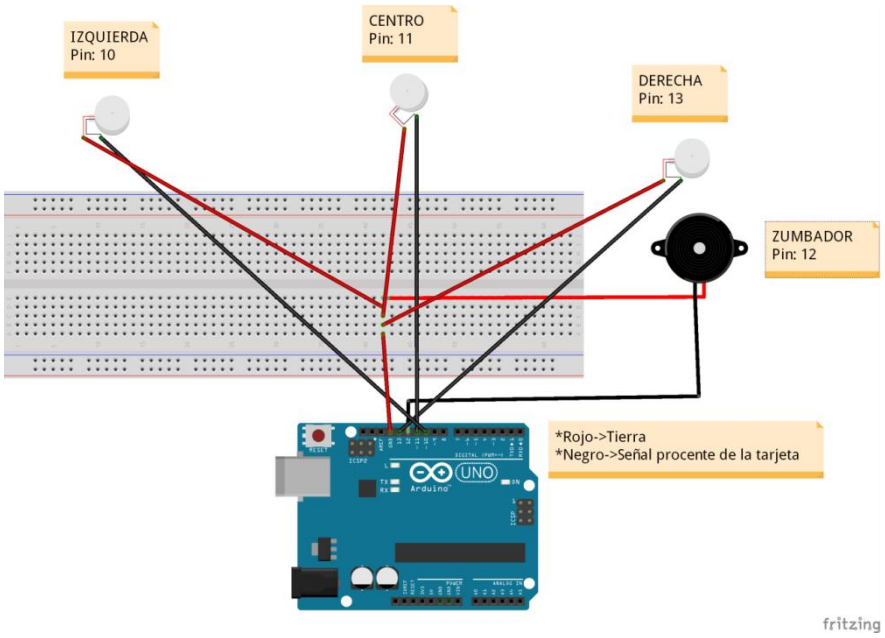

Figura 19. Esquema de conexión de los vibradores y el zumbador

En cuanto al esquema, aclararemos que a cada uno de los sensores de ultrasonidos, le debe llegar una Vcc (de 5V) y una tierra, se denominarán como izquierda, centro y derecha respectivamente. Además les llegarán dos cables, uno en el que les enviarán la señal para realizar el pulso ultrasónico y otro con el que enviarán la información de recepción del pulso ultrasónico que excita el sensor, de vuelta. Los puertos serán para la derecha: El puerto 7 de recepción y 6 de envío. Para el del centro: El 9 para recepción y el 8 para envío. Y por último el izquierdo utilizará el 5 para recepción y el 4 para envío.

Respecto a los excitadores, tendremos en el pin 12 el envío de señales al zumbador, y en los puertos 10, 11 y 13 los vibradores que estarán relacionados con los ultrasonidos: izquierdo, centro y derecho respectivamente. Por último todos ellos deberán tener una patilla que les lleve a tierra.

Los estímulos, en principio, intentaremos hacerlos en modo de secuencia, diferenciando entre tonos distintos para cada sensor que acelerarán su frecuencia en cuanto a la distancia. En el caso de los vibradores, activaremos cada uno de ellos dependiendo del sensor que esté por debajo de la distancia de umbral, lo único que cambiaremos será la frecuencia con la que se emite, dependiendo del mismo modo de la distancia a la que se encuentre el obstáculo.

Todo esto tiene que ser comprobado experimentalmente con personas, para que nos aconsejen que cosas debemos mejorar y de qué modo es más fácil orientarse.

Seguidamente, tenemos que tener en cuenta el modo de programar, porque Arduino no tiene interrupciones. Esto es una gran desventaja porque no pueden

utilizarse funciones ni bucles que sean bloqueantes, debido a que los sensores deben estar todo el tiempo midiendo y tenemos que saber cómo solventar esta desventaja.

Para concluir mencionar que después de haber comprobado el funcionamiento de los ultrasonidos, tenemos que calcular la distancia con la fórmula:

$$distancia(cm) = 0.17cm/s * tiempo$$

Fórmula 1.Cálculo de distancia para los sensores

Tiempo, es el tiempo que tarda en devolver la señal de ultrasonidos desde que se envió el impulso eléctrico al sensor, el valor de 0.17, es la velocidad en centímetros por segundo a la que van las ondas del sensor, aproximadamente. La distancia como ya hemos indicado vendrá dada en centímetros, por lo tanto deberemos tenerlo en cuenta para los umbrales que utilicemos en el código.

2.4 Explicación Código uso zumbador

Código:

```
#include <Time.h>
long distancia=0;
long distancia1=0;//variables para calcular según lo que
devuelve el sensor
long distancia2=0;
long distancia3=0;
long tiempo=0;
int toca; //variable que sirve para indicar el ultrasonido
que hay que leer
//función que reproduce la secuencia correspondiente
void ControlAudio(int tipo, int pausa){
    static long int tiempo_ultima_vez_audio = millis();
    static bool primero_emitido = false, segundo_emitido =
false;
    if(tipo>=0){
        long int tiempo = millis();
        long int sonando = tiempo-tiempo_ultima_vez_audio;
        if(sonando<100){
            if(!primero_emitido){
                //emitir primer tono
                switch(tipo){
                case 0: tone(12,440,100); break;
                case 1: tone(12,300,100); break;
                case 2: tone(12,200,100); break;
                }
                primero_emitido=true;
            }
        }else if(sonando<200){
            //esto es pausa
        }else if(sonando<300){
            if(!segundo_emitido){
                //emitir segundo tono
                switch(tipo){
                case 0: tone(12,440,100); break;
                case 1: tone(12,300,100); break;
                case 2: tone(12,200,100); break;
                }
                segundo_emitido = true;
            }
        }else if(sonando<450+pausa){
            //esto es una pausa
        }else{
            primero_emitido = false;
            segundo_emitido = false;
            tiempo_ultima_vez_audio = tiempo;
        }
    }
}
void leer(int a,int b){
     digitalWrite(a,LOW); /* Por cuestión de estabilización
    del sensor*/
    delayMicroseconds(5);
    digitalWrite(a, HIGH); /* envío del pulso ultrasónico*/
```

```
    delayMicroseconds(10);
    tiempo=pulseIn(b, HIGH);
    delayMicroseconds(10);
    distancia= int(0.017*tiempo); //cálculo de la distancia
    switch(a){
    case 9:distancia1=distancia;break;
    case 7:distancia2=distancia;break;
    case 5:distancia3=distancia;break;
    }

    long dmin = distancia1;
    int pmin = 0;

    if(distancia2<dmin){
        dmin = distancia2;
        pmin = 1;
    }

    if(distancia3<dmin){
        dmin = distancia3;
        pmin = 2;
    }
    if(dmin<200){
        int pausa;
        pausa=99*pow(1.01,dmin);
        ControlAudio(pmin,pausa);
    }

    return;
}
void setup() {
    Serial.begin(9600);
    toca = 0;
    Serial.println("Inicializacion");
    pinMode(9, OUTPUT);
    pinMode(8, INPUT);
    pinMode(7, OUTPUT);
    pinMode(6, INPUT);
    pinMode(5, OUTPUT);
    pinMode(4, INPUT);
    pinMode(12, OUTPUT);
}
void loop() {
    switch(toca%3){
    case 0: leer(9,8);break;
    case 1: leer(7,6); break;
    case 2: leer(5,4); break;
    }
    toca++;

    Serial.println(" distancia1");
    Serial.print(distancia1);
    Serial.println(" distancia2");
    Serial.print(distancia2);
    Serial.println(" distancia3");
    Serial.print(distancia3);
}
```

Explicación:

Para comenzar hemos hecho una programación lo más modular posible reduciendo el código al máximo número de funciones. De esta manera será más sencillo detectar errores y más fácil la comprensión del funcionamiento.

- Bucle principal del programa (**loop**):

  En primer lugar, para el código principal, debemos comentar que teníamos que desarrollar un modo de medida que fuese ágil y rápido, para ello, concebimos la idea de medir 1 ultrasonido por cada iteración del bucle principal que ocurriese, con ese fin, se utiliza un switch, el cual depende del resto de la variable **"toca"** al dividirlo entre 3, y de esa manera según el resultado, crear tres casos, los cuales llaman a la aplicación leer, pasándole los número de los pines a los que los sensores están conectados directamente como argumento.

- Inicialización de las variables (**setup**)**:**

  La función **Setup** es propia de Arduino y es la primera que compila en orden de inicializar todas las variables con un valor.

  En segundo lugar asignamos un valor de 9600 a la tasa de bits con **Serial.Begin**

  Por último con la función **pinMode** identificamos como salida (OUTPUT) o entrada (INPUT) cada uno de los pins, para ello se ha tenido en cuenta el esquema de la ***Figura 14.***

- Función **leer:**
  Esta función tiene como misión principal enviar los impulsos eléctricos que activan los sensores, además de interpretar las señales procedentes estos, calculando la distancia.
  En primer lugar con la función **digitalWrite**, se estabiliza el sensor enviando un pulso, para ello utiliza el valor del pin **a**, que en este caso dependiendo del sensor que toque será 9,7 o 5, y después de un delay de 5 microsegundos enviará el pulso definitivo que se utilizará para medir.

  En segundo lugar con la función **pulseIn**, y el pin **b** (con valor 8,6 o 4 según sea el sensor que toque leer) mide el tiempo que ha tardado en recibir el impulso de vuelta por parte del ultrasonido.
  Posteriormente calcula la distancia con la ***Fórmula 1*** antes citada, y asignará a la **distancia1**, **distancia2** o **distancia3** dependiendo del pin que se haya utilizado para medir.

En último lugar, deberá comprobar con varios **if** qué distancia es la menor y así, almacenar en **pmin** y **dmin** el puerto del sensor a menor distancia y el valor de ésta respectivamente. Por ultimo calculará la pausa que introduciremos entre los tonos que suenen con la ***Fórmula*** 2 (la función ha sido calculada previamente, como una función exponencial para que tenga como distancia mínima un valor máximo igual a **2m** que sume un retardo de **500ms** y un valor mínimo de **1cm** que sume **100ms**), y además de esta última, enviará el **pmin** a **ControlAudio** como argumentos, la función que modifica los tonos y el espacio entre ellos controlando el zumbador.

$$\boldsymbol{pausa = 99 * 1.01^{dmin}}$$

Fórmula 2. Función exponencial que calcula la distancia

Por último se ha tenido en cuenta una función exponencial del tipo: **$x(y)=a*r^y$** para calcular la antes mencionada pausa, para ello debemos saber los valores máximos y mínimos de x e y que serán 200 y 1 y 500 y 100. El primer paso es calcular la **r**, para ello debe sustituirse la fórmula en los valores 200, y 1 quedando así:

$$\frac{x(200)}{x(1)} = \frac{a * r^{200}}{a * r^{1}} = r^{199} = \frac{500}{100}$$

*Fórmula 3. Cálculo del parámetro **r***

Después de este paso, podemos sacar el valor de **r**, el cual nos da1.01 aproximadamente. En cuanto al cálculo de **a**, sólo hay que sustituir, r en la fórmula, en uno de los puntos que sepamos.

$$x(1) = a * 1.01^{1} \approx a * 1.01 = 100$$

*Fórmula 4. Cálculo del parámetro **a***

Por lo tanto obtenemos el valor correspondiente a **a,** que en este caso corresponde con 99. Finalmente obteniendo la ***Fórmula 2***, para la distancia.

Como podemos observar en la **Figura 18,** los valores que buscamos obtener como máximos y mínimos se cumplen, y podemos hacernos una idea de cómo variará la pausa conforme a la distancia para que lo note el usuario.

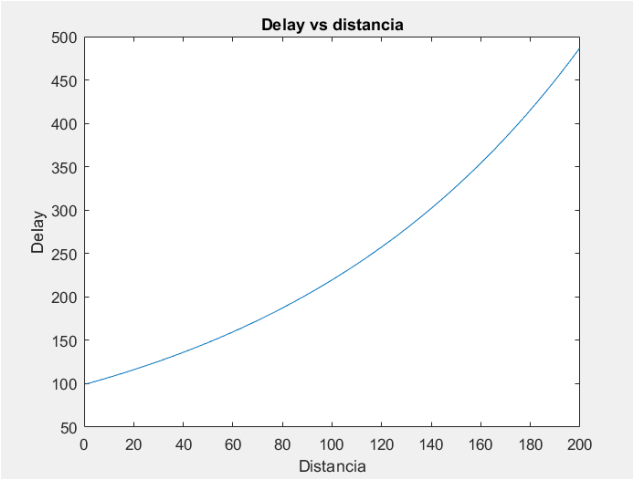

*Figura 20. Representación de la **Función 2** con el software Matlab*

- Función **ControlAudio:**
  Esta función se encargará de realizar la secuencia de tonos que debe producir el zumbador, dependiendo del sensor que excita y esté más cerca, además de la frecuencia con la que debe repetirse para indicar la proximidad del obstáculo.

  Para empezar, esta es una de las funciones que más dificultad tienen, debido a la imposibilidad de trabajar en Arduino con interrupciones, ya que no puede haber ninguna función bloqueante. Lo primero que comprobamos, es que la función **tone,** no es bloqueante, por lo tanto la utilizaremos sin temor a que ralentice la velocidad de lectura de los ultrasonidos.

  La secuencia será dos tonos que duran 100ms, los cuales estarán separados por una pausa de 100ms, y el tiempo que hay entre el comienzo de un nuevo ciclo y el final de éste, tendrá que durar **450+pausa** milisegundos.

  En cuanto al código, en la primera se obtiene el tiempo, para poder manejarlo y saber cuándo tenemos que emitir cada orden. Así que para ello nos valimos de la librería **Time.h,** la cual contiene una función llamada **millis,** que devuelve el tiempo en milisegundos que hace desde cuándo se compiló el programa, es por ello que el tiempo lo almacenaremos en la variable **tiempo_ultima_vez_audio,** que es del tipo **static long int.**

  Además, crearemos dos variables booleanas que nos informarán si se ha activado el audio, para que no se paren y se produzcan nuevos tonos,

cada vez que se está dentro del tiempo que duran cada uno de los intervalos en los que hay un tono, estas también serán del tipo **static.** Se llamarán **primero_emitido** y **segundo_emitido** haciendo referencia a los dos tonos que constituyen la secuencia**.**

En la segunda parte del código, para eliminar posibles errores o estados espurios, se comprueba que el pin que nos pasen sea mayor que 0.

Posteriormente dentro de ese condicional mediremos el tiempo con **millis**, esta vez será el tiempo actual, y calcularemos la variable **sonando**, que será la resta de **tiempo** menos **tiempo_ultima_vez_audio**.

Más adelante, tenemos 4 **if** los cuales representan el eje del tiempo.

El primero comprueba que el tiempo sea menor que 100ms; si es así, debe comprobar que sea falso el contenido de **primero_emitido;** en ese caso emitirá el tono con la función **tone**, con la frecuencia correspondiente y con una duración de 100ms; gracias al condicional **switch** con el puerto introducido, y modificará el estado de la variable **primero_emitido** a falso.

En cualquier otro caso en el que **primero_emitido** fuese falso no haría nada esta parte del código.

El segundo condicional principal debe cerciorarse de que el tiempo es menor que 200ms, ya que accederá cuando **sonando** sea mayor que 100ms por la lógica impuesta, este **if** será simplemente una pausa.

La tercera parte es análoga a la primera, con la diferencia de que esta debe funcionar mientras sea menor a 300ms. En este caso cambiará el modo de emitir por otro tono, ya que según se decidió, el sensor del centro produce dos todos de 440Hz, el de la derecha dos tonos de 300Hz, y por último el de la izquierda dos tonos de 200Hz.

Cabe destacar que esto es en una primera aproximación y que no están exentos de cambio, porque la experimentación con varios sujetos, además de nuestro sentido común según avance el proyecto, nos harán decidir si quedarnos con este tipo de secuencias o modificarlas de forma pertinente.

Para culminar esta función, el último de los condicionales comprobará que sea menor de **450+pausa milisegundos**, dicha pausa habrá sido

calculada en la función **leer** y enviada mediante los argumentos cuando se llama esta función.

El número 450 viene dado por la suma de los **100ms** del primer tono, los **100ms** de pausa, los **100ms** del segundo todo, y el mínimo de **150ms** de pausa obtenido gracias a la experimentación, ya que con una pausa menor haría que no sonase correctamente el zumbador. En cuanto este tiempo sobrepase el umbral, se modificará **tiempo_ultima_vez_audio,** asignándole el valor de la variable **tiempo,** además deberán ponerse en **false** el valor de las variables **primero_emitido** y **segundo_emitido,** ya que comenzara otro ciclo a partir de ahí.

Para acabar con el comentario del código, mencionaremos que se declararán como variables globales las variables: **distancia, distancia0, distancia1, distancia2, toca y tiempo.** Todas ellas con un valor de 0.

2.5 Explicación Código uso 3 vibradores

Código:

```
#include <Time.h>
long distancia=0; //variables para calcular según lo que
devuelve el sensor
long tiempo=0;
int toca; //variable que sirve para indicar el ultrasonido
que hay que leer
//funciones que activan cada uno de los vibradores
void ControlVibracion0(int frec){
    analogWrite(11, frec);
    return;
}

void ControlVibracion1(int frec){
    analogWrite(13,frec );
    return;
}

void ControlVibracion2(int frec){
    analogWrite(10, frec);
    return;
}
void leer(int a,int b){
    digitalWrite(a,LOW); /* Por cuestión de estabilización
del sensor*/
    delayMicroseconds(5);
    digitalWrite(a, HIGH); /* envío del pulso ultrasónico*/
    delayMicroseconds(10);
    tiempo=pulseIn(b, HIGH);
    delayMicroseconds(10);
    distancia= int(0.017*tiempo); //cálculo de la distancia
    if(distancia<200){
        Serial.println(" distancia");
        Serial.print(distancia);
        switch(a){
        case 9:
            if(distancia<100){
                ControlVibracion0(150);
            }else{
                ControlVibracion0(75);
            }
            break;
        case 7:
            if(distancia<100){
                ControlVibracion1(150);
            }else{
                ControlVibracion1(75);
            }
            break;
        case 5:
            if(distancia<100){
                ControlVibracion2(150);
            }else{
                ControlVibracion2(75);
```

```
                }
                break;
            }
        }else{
            switch(a){
            case 9:
                analogWrite(11, 0);
                break;
            case 7:
                analogWrite(13, 0);
            case 5:
                analogWrite(10, 0);
                break;
            }
        }

        return;
    }
    void setup() {
        Serial.begin(9600);
        toca = 0;
        Serial.println("Inicializacion");
        pinMode(9, OUTPUT);
        pinMode(8, INPUT);
        pinMode(7, OUTPUT);
        pinMode(6, INPUT);
        pinMode(5, OUTPUT);
        pinMode(4, INPUT);
        pinMode(12, OUTPUT);
        pinMode(10, OUTPUT);  //izquierda
        pinMode(11, OUTPUT);  //centro
        pinMode(13, OUTPUT);  //derecha
    }

    void loop() {
        long int tiempo = millis();
        switch(toca%3){
        case 0: leer(9,8);break;
        case 1: leer(7,6); break;
        case 2: leer(5,4); break;
        }
        toca++;
    }
```

Explicación:

Por no redundar información, en primer lugar debemos comentar que este código es absolutamente análogo al del zumbador. Por ello, se mencionarán únicamente los cambios que son verdaderamente importantes para relatar cómo funciona este código para el prototipo.

- Función **leer:**

  Los vibradores tienen una particularidad, y es que al funcionar uno, no significa que otro no funcione, ya que el zumbador emite con la secuencia del ultrasonido que esté a menor distancia, y en este caso lo que importa, es que, el situado por debajo de dos metros, emita.

  Por ello, después de calcular la distancia de igual forma que hace en el código del zumbador, comprueba qué ultrasonido se estaba leyendo, seguidamente hay un condicional que comprueba que la distancia al obstáculo calculada sea menor a 200 (2m). Entonces en ese condicional tipo **switch**, según el puerto de ultrasonido al que corresponda la distancia, llamará a las funciones que activan los vibradores (**ControlVibración1**, **ControlVibración2 y ControlVibración3),** pasándoles la frecuencia a la que se activan, siendo 75 o 150 el valor analógico introducido a cada vibrador, dependiendo si se encuentra a más de un metros o menos respectivamente. En cualquier otro caso que la distancia sea menor a 2 metros, se apagará el vibrado que acabe de medirse la distancia gracias al **switch** incluido en la parte **else** del condicional.

  Por último, destacar que para que el código fuese más claro, hemos creado 3 funciones llamadas: **ControlVibracion0, ControlVibracion1** y **ControlVibracion2,** para activar los motores vibradores centro, derecha e izquierda respectivamente. Las cuáles, serán llamadas desde cada caso del **switch** antes mencionado.

- **Funciones ControlVibraciónX:**

  Estas funciones escriben el número que les pasen como argumento, (que será 150 o 75, referidos a un rango que va desde 0, que es el valor mínimo, a 255, que es el valor máximo) en el puerto correspondiente a centro, derecha o izquierda que son 11, 13 y 10 respectivamente. Todo ello para hacer oscilar al vibrador.

2.6 Experimentación

La experimentación es una de las claves de este proyecto, ya que sólo de esta manera sabremos que estamos haciendo las cosas de la manera correcta, adecuando los cambios pertinentes para la comodidad y la efectividad del sistema de detección de obstáculos aéreos.

En nuestro caso, hemos realizado esta parte con varias personas ajenas, las cuales pueden, siendo imparciales, opinar del funcionamiento del sistema.

Por otra parte hemos detectado por experiencia propia, qué elementos se deberían cambiar, y muchos de ellos coincidían con las opiniones de los sujetos que han probado el detector.

En primer lugar probamos mediciones para ver si la pausa funcionaba correctamente, porque al variar de una manera tan rápida es muy difícil darse cuenta de la distancia a la que está el objeto, por ello, con el **Serial plotter** graficamos en el caso del zumbador, que es el único que utiliza una fórmula para la frecuencia con la que se repite la secuencia, los datos de distancia y pausa que este nos ofrecía.

En este caso hicimos dos barridos con la mano por delante de los sensores, y como se puede observar, contando las veces que mide el sensor en relación con el tiempo que se ha estado midiendo, tiene una frecuencia de medida de 8.33Hz, o lo que es lo mismo toma una muestra cada 0.12 segundos.

En cuanto a la gráfica, podemos observar la variación de la pausa con los “dos obstáculos” (las dos veces que se pasó la mano).

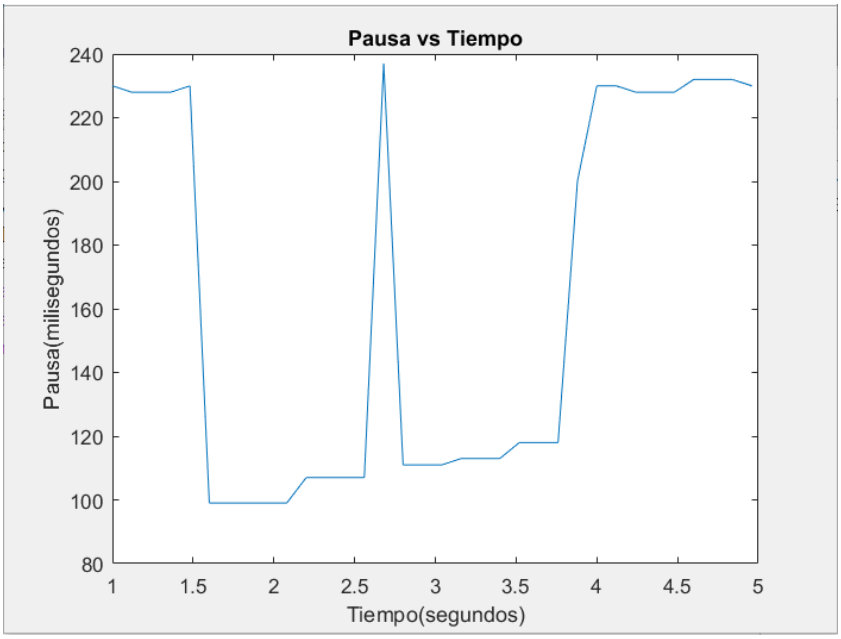

Figura 21. Gráfica Pausa introducida entre la secuencia frente al tiempo

Posteriormente, hicimos que nos sacase por pantalla la evolución de las distancias de los 3 ultrasonidos, a ver qué tal funcionaba, además de para ver la correlación de distancias entre ellos, el resultado fue este:

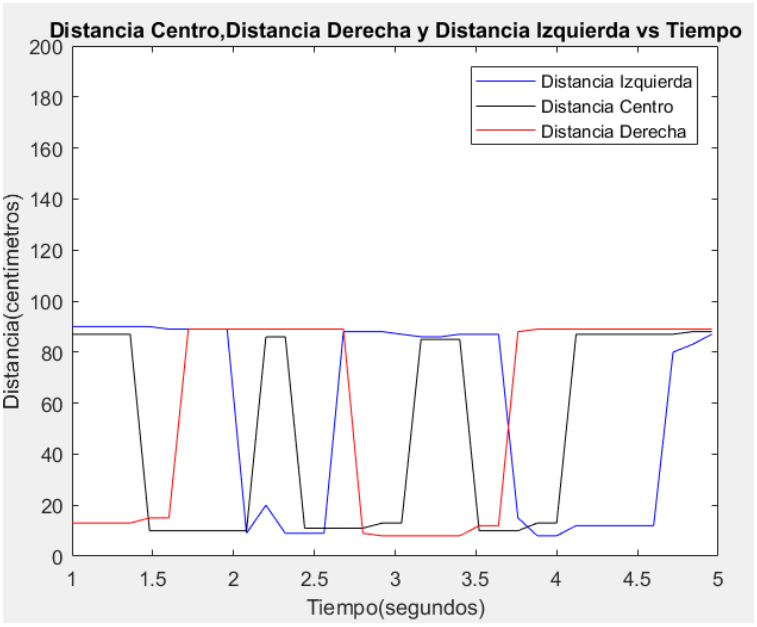

Figura 22. Distancia Centro, Distancia Derecha y Distancia Izquierda enfrentadas contra el tiempo

En primer lugar debemos comentar que hemos realizado 3 barridos con la mano de una forma que no sea lo suficientemente grande como para tapar los tres a la vez y de esta manera en la representación, puedan diferenciarse las etapas.

Como podemos observar, cada vez que baja cada una de las distancias, significa que hay un obstáculo delante del sensor que proceda, es por eso que al estar enfrentados los sensores contra una pared situada a 90cm, más o menos, del sistema, podemos diferenciar cuando se acerca un obstáculo por la disminución de la distancia.

Si nos fijamos, la subida del trazado rojo, comienza a subir cuando baja la del negro, esto es así porque el obstáculo ha pasado desde el lado derecho al centro. Esto, se puede extrapolar a cualquier transición entre los lados.

Para concluir este apartado de la experimentación, cabe destacar que el sistema, además de tener una velocidad de lectura bastante alta, funciona perfectamente con ambos códigos, tanto como en el cálculo de la pausa conforme a la distancia como en la detección de los obstáculos.

La segunda parte de la experimentación, fue probar los códigos con diferentes personas, que sean ajenas a los programadores, para que puedan decirnos, cómo es su experiencia y que aspectos pueden mejorar el prototipo.

Figura 23. Experimentación del prototipo I

Figura 24. Experimentación del prototipo II

Entre las recomendaciones del ámbito de la percepción y físicas sugeridas por los sujetos que probaron el prototipo, hemos destacado las siguientes:

Figura 25. Experimentación del prototipo III

- Los cables no deben tocar la cabeza de la persona, por ello, deben utilizarse bridas, además de que se agrupen de la mejor manera posible. En el caso de que tuviésemos instrumental más avanzado, podría hacerse un circuito en una placa, esto mejoraría la estética, además de dejar un diseño menos complicado.

- En cuanto al sistema que utiliza estímulos sonoros, nos hemos dado cuenta que así funciona muy bien, pero su uso es más intuitivo y tiene menos dificultad, si utilizamos solo una frecuencia para cada punto de medida; es decir para derecha e izquierda en vez de usar una secuencia con tonos de 300 y 200Hz y 200 y 300Hz

respectivamente, cambiar a 300 y 300Hz y 200 y 200Hz, para que se sepa que ese sonido siempre está relacionado con ese lugar.

Además la utilización de tonos diferentes hará que si hay alguna interferencia, o algún error de medida, identifiquemos en qué lugar está el obstáculo de una forma más rápida y mejor para poder esquivarlo.

- En lo referido al uso de estímulos hápticos, después de la experimentación con varios sujetos, decidimos, en primer lugar, cambiar la secuencia para que el vibrador oscile a una frecuencia más alta conforme más cerca se encuentre el obstáculo. De esta manera funcionaría mejor el código, y sería más fácil discernir la procedencia concreta de la vibración.

  Como conclusión de la experimentación, debemos modificar las partes antes mencionadas, es por ello que nos dimos cuenta de lo importante que es este proceso en el desarrollo y el perfeccionamiento de un proyecto, ya que así conseguimos un resultado que de verdad sirve para orientar del posicionamiento de obstáculos aéreos a una persona invidente. En un futuro, nuestro objetivo sería probarlo con gente invidente de verdad, pero debido a la carga del trabajo del curso, sumado al tiempo y dedicación que requiere poder hablar con una asociación para poder trabajar con una persona invidente fue imposible.

2.7 Solución final

Este será el resultado final estético del proyecto:

En primer lugar, debemos posicionar los ultrasonidos con los pines orientados hacia arriba para que los cables no molesten en ningún momento, al rozar la cara del sujeto.

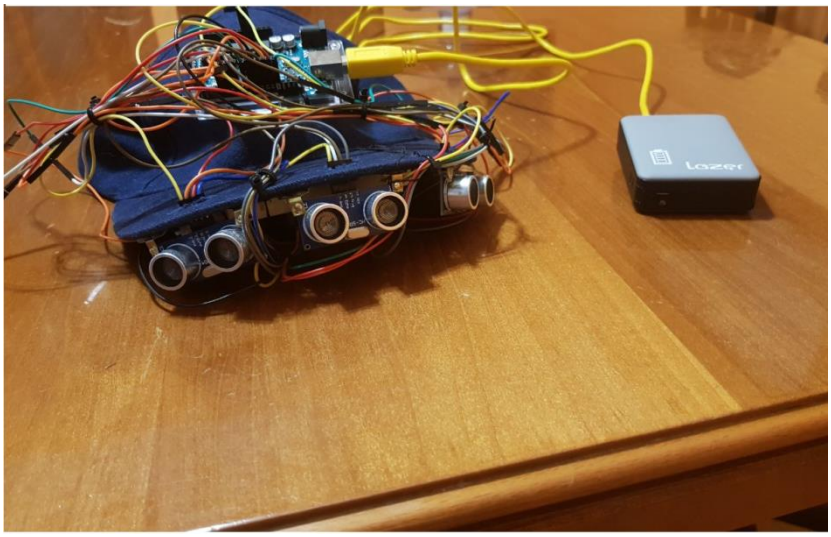

Figura 26. Disposición de los sensores de ultrasonidos

En segundo lugar hemos colocado el zumbador debajo de la visera, en una placa board de tamaño reducido, cosida en la visera, situada en la parte trasera detrás de los sensores.

En tercer lugar, hemos colocado bridas para poder juntar lo más posible los cables que van hacia la placa board y la tarjeta de Arduino.

Así que concluyendo este es el resultado final de cómo quedarían los cambios sugeridos hechos:

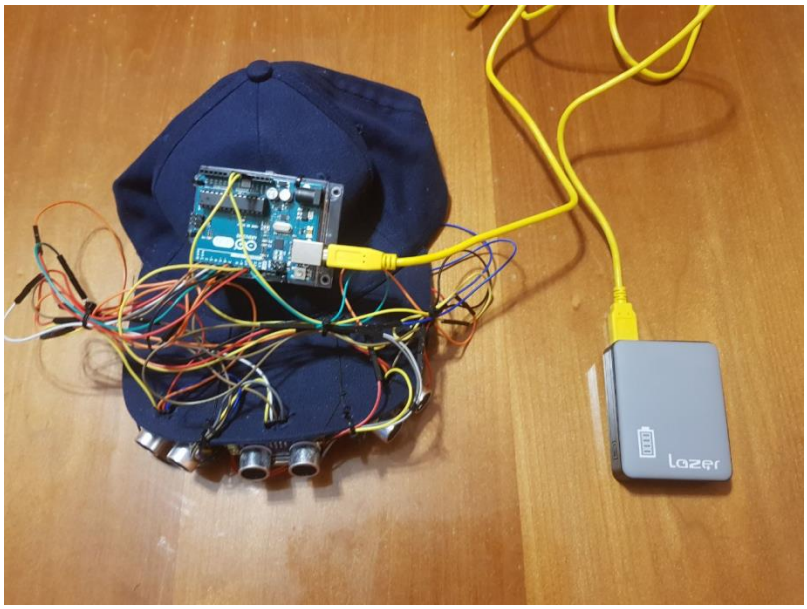

Figura 27. Prototipo finalizado I

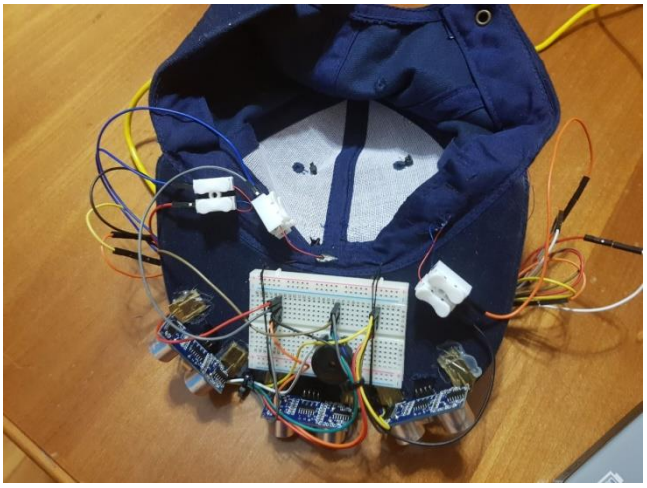

Figura 28. Prototipo Finalizado II

3. Conclusión

En primer lugar cabe destacar, el buen resultado que se ha obtenido con este trabajo, ya que hemos conseguido, pese a un arduo trabajo, idear tanto la estructura, como el código de un sistema que además de utilizar 2 tipos de estímulos, explota lo eficiente económicamente y lo potente que puede llegar a ser Arduino.

En segundo lugar, saber, que partiendo de esta solución, abre un mundo de desarrollo y de posibles nuevos proyectos que puedan, basándose en nuestro proyecto, hacer una solución que poco a poco integre más a los discapacitados visuales, en la sociedad. Es por ello que el feedback de internet, en una comunidad que no deja de desarrollarse y crecer, es un punto importantísimo en este proyecto, porque esto no queda solo aquí, como he mencionado antes, el futuro que tiene este tipo de soluciones es abrumador, y cada vez, está más al alcance de todo tipo de personas.

En tercer lugar haremos una síntesis económica sobre el proyecto, comenzaremos por hacer un listado de los componentes necesarios y su precio para que cualquier persona a la que le dejemos nuestros códigos, sea capaz de fabricárselo, sin tener un amplio conocimiento tecnológico o sobre la programación.

Elementos necesarios	Cantidad	Precio Unitario
Sensor de ultrasonidos HC-SR04	3	3€
Motor vibrador	3	5.5€
Kit básico que contenga la tarjeta Arduino UNO con al menos un zumbador y cables	1	35.90€
Kit cables Arduino variados(macho-macho, macho-hembra, hembra-hembra)	1	6€
Gorra	1	3€
Total		70.4€

Tabla 1. Elementos necesarios para construir nuestra solución

Como podemos ver el coste real de realización de la solución serían unos 70€(el precio puede variar según la tienda), ya que solo deberían instalarlo en una gorra básica, además de descargarse nuestro código. Como parte complementaria del proyecto, sería posible subir un video o un esquema sobre cómo montarlo, para que cualquiera pueda llevar a cabo este sistema.

Por lo tanto en este apartado obtenemos un rotundo éxito, rompiendo la barrera de producción para un producto, que es muy necesario para el colectivo de discapacitados visuales, poniéndolo a disposición a un precio que dista mucho de los 300 dólares de la Sunu band, de las 635 libras del UltraCane o de los 2420€ del Eyesynth por ejemplo.

Las tiendas donde hemos adquirido los componentes son **Bricogeek** (https://tienda.bricogeek.com) y **Cetronik** (http://www.cetronic.es/)

Por último concluir diciendo, que si para un futuro, deberíamos mejorar la estética el volumen y la autonomía del prototipo, que por lo tanto serían los nuevos objetivos en un futuro trabajo.

4. Bibliografía

- J.M. Sáez, F. Escolano, M.A. Lozano, "Aerial obstacle detection with 3D mobile devices", IEEE Journal of Biomedical and Health Informatics, 2014.
- J.M. Sáez, F. Escolano, "Stereo-based Aerial Obstacle Detection for the Visually Impaired", European Conference on Computer Vision (ECCV), Marselle, France, October 2008.
- Brian W. Evans, "Arduino Programming Notebook: A Beginner's Reference", 2008.
- Massimo Banzi, "Getting Started with Arduino", 2009.
- El cajón de Arduino (http://elcajondeardu.blogspot.com/)
- Youtube como plataforma con variedad de tutoriales (https://www.youtube.com/)
- Foro de ayuda de Arduino (https://forum.arduino.cc/)
- Web oficial de Arduino (https://www.arduino.cc/)
- Sunu band (https://www.sunu.io/index.html?)
- Sonar Globe (http://www.instructables.com/id/Sonar-Glove-for-the-Visually-Impaired/)
- Eyesynth (https://eyesynth.com/que-es-eyesynth/)
- DOA (http://rua.ua.es/dspace/handle/10045/37205)
- UltraCane (https://www.ultracane.com/)